A limited discrepancy search method for solving disjunctive scheduling problems with resource flexibility

Marie-José Huguet, Pierre Lopez, Abir Ben Hmida

A Limited Discrepancy Search method for solving disjunctive scheduling problems with resource flexibility

Huguet M.-J.[1,2], Lopez P.[1], and Ben H'Mida A.[3]

[1] LAAS-CNRS, 7 av. du colonel Roche, 31077 Toulouse, France
e-mail: {huguet,lopez}@laas.fr
[2] INSA, 135 av. de Rangueil, 31077 Toulouse, France
[3] SOI^{2}E, Institut Supérieur de Gestion, Tunis, Tunisia
e-mail: abir.ben-hmida@laposte.net

Keywords. Limited Discrepancy Search, Heuristics, Scheduling, Resource Flexibility.

1 Problem statement

In this paper we are concerned with shop scheduling problems with resource flexibility. We propose to solve both problems, scheduling and allocation, in an integrated approach based on an adaptation of Limited Discrepancy Search (LDS) (Harvey and Ginsberg (1995)) to prune dramatically the search tree while ensuring good performances.

The problem under consideration can be stated as follows. A set T of n tasks is to be performed by a set R of m machines. Every task is processed without interruption within a time window by exactly one machine. Two tasks competing for the same machine must be sequenced. The resources are flexible; this means that to each task $i \in T$ is associated a set $R_i \subseteq R$ of machines able to process it; the task duration, denoted by $p_{i,\mu}$, depends on machine $\mu \in R_i$ on which the task is executed. To a task $i \in T$ are associated three variables: its start time st_i, its finish time ft_i, and its allocated resource $alloc_i$.

For a task i, one has $alloc_i \in R_i$, and $est_i \leq st_i \leq lst_i$ and $eft_i \leq ft_i \leq lft_i$ where est_i and lst_i (respectively eft_i and lft_i) represent the earliest and the latest start time (respectively the earliest and the latest finish time). It is also of interest to introduce rd_μ which denotes the release date of machine μ, that is, the time at which the machine is available to process a task. Moreover certain tasks may be linked together by general precedence constraints. Finally, the objective is to minimize the makespan: $\min(\max_{i=1..n} ft_i)$. This problem is NP-hard since some other related optimization scheduling problems (multi-purpose machines and multi-mode resource constrained scheduling problems) are known to be NP-hard (Brucker et al. (1997), Heilmann (2003), Kolisch (1995), Mastrolilli and Gambardella (2000)).

2 Limited Discrepancy Search

The objective of Limited Discrepancy Search (Harvey and Ginsberg (1995)) is to provide a tree-search method for supervising the application of variable and value ordering heuristics. To understand the principle of LDS, let consider a set of instantiations (*i.e.* a *path*) given by a heuristic. Knowing that this path ends either at a dead end or at a goal node, iteration k of the procedure (denoted by LDS(k))

fixes a number k of decision points to contradict over the decisions taken by the heuristic. Such a contradiction point is called a *discrepancy*.

The LDS method was developed for binary problems. In this case, for N binary decision variables the number of leaves is 2^N, the maximum number of discrepancies is N, and the number of paths with exactly k discrepancies is C_N^k.

An important limitation of LDS is that, for a given number k, all paths from 0 up to k discrepancies are enumerated when applying LDS(k). Thus, many redundancies are encountered when k varies from 0 to C_N^k. To remove these redundancies, an *improved LDS* (ILDS) has been proposed in Korf (1996): in ILDS(k) only the paths with exactly k discrepancies are evaluated. Another extension aiming at improving the original LDS is termed *Depth-bounded Discrepancy Search* (DDS) (Walsh (1997)). The motivation behind DDS is to make discrepancies at the top of the search tree since decisions at a top level have an important impact on heuristic failings.

Many applications of LDS exist in the field of disjunctive scheduling (see for example Caseau (1997), Harvey and Ginsberg (1995), Jackson et al. (1995)). It is easy to model these problems with binary decisions: a 0-1 variable corresponds to sequence a pair of tasks in a given order and the associated search tree is also binary.

3 Our LDS Method

The concept of discrepancy is redefined so as to suit to flexible scheduling problems in which decision variables are discrete. Indeed, considering a set of ordered discrete values $\{v_0, v_1, \ldots, v_k\}$ a variable x_i can take, it seems natural, as in Loudni and Boizumault (2001), to associate a 0-discrepancy to value v_0, a 1-discrepancy to value v_1, ..., a k-discrepancy to value v_k. This allows us to limit the tree search by giving a priority to an assignment as low as its rank is high in the list given by the ordering heuristic.

For N decision variables such that the number of discrepancies for each variable x_i is from 0 to d_i, the number of leaves of the search tree is equal to $\prod_{i=1}^{N}(d_i+1)$; and the maximal number of discrepancies is equal to $\sum_{i=1}^{N} d_i$. The number of paths with exactly k discrepancies is given by the coefficient of term x^k in the polynomial $\prod_{i=1}^{N}(\sum_{j=0}^{d_i}(x^j))$.

With this new definition of discrepancy, we propose to adapt the LDS method for the problem under study. To do that, we have to explain how to obtain an initial solution and how the tree search is expanded by way of discrepancies.

Heuristics for the computation of an initial solution Here, our goal is to find a solution with a minimum makespan by taking simultaneously into account scheduling and allocation variables. To face these problems, we propose:

1. to select a task, for instance i;
2. to assign this task to a resource, for instance $alloc_i \leftarrow k$;
3. to start this task as soon as possible knowing the resource allocated for its processing: $st_i \leftarrow \max(est_i, rd_k)$ then $ft_i \leftarrow st_i + p_{i,k}$;
4. to propagate (by forward checking) these decisions on not yet instantiated variables: $rd_k \leftarrow ft_i$ and $est_j \leftarrow \max_{j \in Succ_i}(est_j, ft_i)$ ($Succ_i$ = successors of i);
5. and to re-start to the first step.

This process leads us to a solution: there is no dead end such as with the classical LDS method since each decision is propagated and the task finish times are not constrained (we may always find a solution by delaying a task).

We propose to consider two task ordering heuristics, called *min-est-min-Margin* and *min-est-max-tail*. Heuristic *min-est-min-Margin* gives the priority to tasks with the minimum earliest start time, then with the minimum Margin, then with the minimum number of possible resources (if necessary, task index is used to break ties). In the heuristic *min-est-max-tail*, only the second criterion has been modified. The priority is given to a task with the minimum earliest start time and then with the maximum tail. These two heuristics are static. The task ordering is computed once, before starting the tree search procedure.

For every task, heuristic *min-length-min-dem* gives the priority to the fastest resource, then to the less required one. Heuristic *min-dem-min-length* gives the priority to the less required resource then to the fastest one. Both heuristics are static. A common drawback is that the fastest resource may be not available (this forces to delay the task). To prevent this drawback, heuristic *min-finish-time* gives the priority to the resource such that the task completes as soon as possible. This latter heuristic is dynamic: the resource with the highest priority depends on the resources previously allocated.

Tree search expansion We propose to do not apply discrepancies to all decision variables to limit the tree search expansion. First of all, we only need to consider discrepancy on allocation variables since the start time is fixed as soon as possible according to the selected resource. Moreover, it seems interesting to define a heuristic to select some allocation variables which may improve the makespan of the solution. This heuristic for applying discrepancies then depends on the resource ordering used to find the initial solution.

With the heuristic *min-finish-time*, some tasks of the initial solution may be assigned to a resource which is not the fastest one (in terms of processing time). This is the case when the fastest resource has been previously allocated to another task (which had a higher priority). We then consider that a discrepancy may consist to assign the task with the higher priority to another resource. This heuristic for applying discrepancies is based on the same idea of DDS proposed in Walsh (1997) which consists to apply discrepancies at the top of the search tree.

Our LDS algorithm For a given number k of allowed discrepancies, our algorithm computes solutions with $0, 1, \ldots, k$ discrepancies. The algorithm stops when the number of allowed discrepancies is reached, when there is no more feasible discrepancy (*i.e.* there is no way to improve the set of current solutions with the heuristic for discrepancies), or when an optimal solution is reached.

4 Experiments

The experiments are based on randomly generated instances which have the following characteristics:

- n: the number of tasks;
- $m = \sqrt{n}$: the number of machines;

- $P_{max} = m.(m-1)$: the maximum number of precedence constraints;
- $R_{max} = \max_{i \in T} |R_i|$: the maximum number of resource alternatives per task.

For each value of n (from 9 to 81) and R_{max} (2, 3, 4 or 5) we randomly generate 50 problems and we report the results in average on these problems.

In the first part of experiments, we only compare heuristics for task and resource ordering in terms of makespan and of computation time. The most interesting heuristic for making allocation decision is the dynamic one, *min-finish-time*. This heuristic integrates both time constraints and resource constraints.

In the second part of experiments, the objective is to evaluate the heuristic proposed for applying discrepancies. We then compare the best algorithm for the initial solution in which we consider the heuristic based on *min-finish-time* for applying discrepancies, with another algorithm, based on the same reasoning for task and resource ordering heuristics, but with use of discrepancies for all variables as in the classical LDS method. At the end of our algorithms, we note the makespan of the best solution and the number of discrepancies needed to reach it. The best solution is obtained with few discrepancies and we can also note that there is a short difference between the makespan of the initial solution and the makespan of the best solution found by each of our algorithms. Hence it seems that our heuristics for task and resource ordering are powerful since few updatings lead to a good solution. Then, the heuristic for applying discrepancy does not have a large impact to the value of the makespan of the best solution. But, the algorithm based on the heuristic for applying discrepancy always find the best solution in the shortest CPU time.

References

BRUCKER, P. and JURISCH, B. and KRAEMER, A. (1997): Complexity of scheduling problems with multi-purpose machines. *Annals of Operations Research, 70*, 57–73.

CASEAU Y. (1997): Using constraint propagation for complex scheduling problems managing size. In: G. Smolka (Ed.): *CP'97 – Principles and Practice of Constraint Programming*. LNCS, 1330, Springer-Verlag.

HARVEY, W.D. and GINSBERG, M.L. (1995): Limited Discrepancy Search. *Proc. of the 14^{th} IJCAI*, Montréal, Canada, 607–613.

HEILMANN, R. (2003): A branch-and-bound procedure for the multi-mode resource-constrained project scheduling problem with minimum and maximum time lags. *European Journal of Operational Research, 144*, 348–365.

JACKSON, W.K. and IRGENS, M. and HAVENS, W.S. (1995): A re-examination of limited discrepancy search. Intelligent Systems Lab, Centre for Systems Science. Simon Fraser University, Burnaby, B.C., Canada.

KOLISCH, R. (1995): Project scheduling under resource constraints. Physica Verlag, *Production and logistics*, Heidelberg.

KORF, R.E. (1996): Improved Limited Discrepancy Search. AAAI/IAAI, Vol. 1, 286–291.

LOUDNI, S. and BOIZUMAULT, P. (2001): Une méthode hybride pour la résolution des VCSP en contexte anytime. *Proc. of JNPC'2001*, Toulouse, France, 173–184.

MASTROLILLI, M. and GAMBARDELLA, L.M. (2000): Effective Neighbourhood Functions for the Flexible Job Shop Problem. *Journal of Scheduling, 3*, 3–20.

WALSH, T. (1997): Depth-bounded Discrepancy Search. *Proc. of the 15^{th} IJCAI*, Nagoya, Aichi, Japan, 1388–1395.